CSE 232 Spring 2018

COMPUTER PROJECT #8

Assignment Overview
In this assignment you will practice creating our own user-defined data structures to be used in a simple stock market simulation. You will utilize what you have learned to date as well as using structs. It is worth 60 points (6% of the overall grade). It is due 3/26, Monday, before midnight on Mimir.

Background
We will use some real data from the Dow Jones Industrial Average (http://en.wikipedia.org/wiki/Dow_Jones_Industrial_Average), a set of 30 stocks that are used as an indication of the U.S. stock market. This sample is from 8/31/2012 to 6/14/2001.

Market
You will help create a `Market` struct. The `proj08_market.h` file provides three data members:

- `string file_name` , the name of the file containing the data
- `vector<string> symbol_list`. A list of the short names of each of the 30 stocks in the Dow
- `map<long, vector<double> > stocks`. The prices of the 30 stocks on the date (the key). The prices are in order of the elements in symbol_list (that is, alphabetically by short name)

You must write the following methods:

- a constructor that takes a single string argument, a string file name
  - o That file contains closing stock prices for 30 stocks of the Dow Jones for a particular date (see format below). It populates the data member `stocks` .
- a method `double get_price(string stock, long date).`
  - o returns the price of the stock on the date if:
    - ▪ the date is a valid date
    - ▪ the stock symbol is a valid stock symbol
  - o returns a -1.0 otherwise

- a method `high_low_year(long year, string symbol)`. Returns as a `pair` the high and low values (in that order) for that stock for the provided year.
  - o if the year or the symbol does not exist, returns the pair {-1.0, -1.0};

Player
You will help create a `Player` struct. The proj08_player.h provides 2 data members:

- double cash How much money the player has
- map<string, long> stocks. The key is the stock symbol and the long is the quantity of that stock that the player owns (an integer value of stocks)

You must write the following methods:

- a constructor that takes a single parameter, the `double cash` the player starts with.
- a method `bool buy(Market &m, string stock, long date, long quantity)`. An attempt to buy a stock by the player from the `Market` on the specified date.
  - o returns true if the player:
    - ▪ has enough cash to make the purchase
    - ▪ the stock symbol is one of the valid 30 symbols
    - ▪ the date is valid (within the range of dates stored in Market)
  - o if true, purchase is made and the player info is updated

        - cash reduced, stocks updated
    - if false return, no action taken
- a method `bool sell (Market &m, string stock, long date, long quantity)`
    - returns true if the player:
        - has the stock to sell (can't sell what you don't have)
        - has at least the quantity indicated (can't sell more than you have)
    - if true, player info is updated
        - cash is increased, stocks updated
    - if false, no action taken
- a method `string to_str()`
    - returns a string representation of the player
    - format is: `cash,symbol:quantity,symbol:quantity`… (see Mimir format)
    - numeric output is `fixed, setprecision(2)`
    - always prints the cash value (even if 0.00), but only prints `stocks` (symbol:quant) if there are indeed any key:value pairs in `stocks`.
- a method `Player combine(Player &b)`
    - returns a new `Player` that has, as a combination, all the `cash` and `stocks` of the two players: the caller and the argument.
    - the caller and the argument Players have their `cash` set to 0 and their `stocks` cleared

Requirements
We provide the following three files:
- `proj08_player.h`, the class declaration
- `proj08_market.h`, the class declaration
- `proj08_main.cpp`, a sample to use for testing

You will write the two files:
- `proj08_player.cpp`
- `proj08_market.cpp`

We will test your files using Mimir, as always.

Deliverables
`proj08/proj08_player.cpp` and `proj08/proj08_market.cpp`.
1. Remember to include your section, the date, project number and comments.
2. Please be sure to use the specified directory and file name.
3. You can zip the directory with the two files inside and upload to Mimir.

Assignment Notes
- Format of `dow.txt`. I created this file from the data on the website `http://www.optiontradingtips.com/resources/historical-data/dow-jones30.html` so that each line consists of 31 space separated values. An example is the first line (the below is all actually on one line, but we break it up into multiple lines for readability).

```
20120831   8.53   58.10   71.40    7.98   84.82   18.94  112.16   49.75   49.47   20.55
 56.75   16.75  194.85   24.83   67.43   36.87   41.21   37.15   89.49   92.60   42.64
 30.82   23.86   66.64   36.22   64.28   79.85   42.47   72.60   87.30
```

    - The first value is a date that can be read in as an integer. 20120831 is interpreted as 08/31/2012 (August 31st 2012)
    - subsequently each value on the line can be read as a floating point number. The 30 values represent,

in alphabetical order by stock symbol, the closing value for that stock on that date.

- for the stock symbols and their associated companies, see the file `notesOnDow.txt`.